布比区块链系统接口规范

接口规范

文件状态： [ ] 草稿 [ ] 正式发布 [√]正在修改	文件标识：	BUBICHAIN
	当前版本：	2.0.0.2
	作　　者：	布比
	完成日期：	

布比（北京）网络技术有限公司

Bubi Technologies Limited

版本修订历史

版本/状态	作者	参与者	起止日期	备注
2.0.0.0			2015/8/26	描述修改和相关说明内容
2.0.0.1	赵正涌		2015/10/8	获取交易信息接口增加调用方式。
2.0.0.2	罗荧彬		2016/11/22	文档结构修改，内容改动

目 录

1 文档介绍

1.1 文档目的

描述文档目的。

1.2 文档范围

描述文档包含范围。

1.3 术语与缩写解释

在后续文档内容中，将通过一下简写来替代对应的对象名称描述：

缩写、术语	解 释
TCP	传输控制协议 (Transmission Control Protocol)
HTTP	超文本传输协议 (Hyper Text Transfer Protocol)
SSL	安全套接层 (Secure Socket layer)
TLS	传输层安全 (The Transport Layer Security)
XXX	说明

2 接口数据规范

接口通信方式基于 TCP，采用 HTTP/1.1 协议(RFC2616)，一次接口调用包含一个完整的请求和响应，请求和响应中均可以携带相应的数据。

接口内容包括接口名称、HTTP 方法、请求内容和响应内容，定义如下：

1. 接口名称为 HTTP URI 名称，如：interfaceName。
2. 接口请求和响应携带数据内容采用 **XML**。
3. 字符编码采用国际化字符编码集 **utf-8**。
4. 携带 XML 数据内容对字符大小写敏感（区分大小写）。
5. 接口调用结果通过 HTTP 状态码标识，200 表示成功，其他代码全部表示错误，通过 HTTP 描述信息描述出此次错误的信息。

2.1Post

示例请求：

```
GET /interfaceName?param1=value1&param2=value2 HTTP/1.1\r\n
Content-Type: application/xml\r\n
Content-Length: xxx\r\n
\r\n
```

示例响应：

```
HTTP/1.1 200 OK\r\n
Content-Type: application/xml\r\n
Content-Length: xxx
\r\n
{"error_code":0, "errorMessage":"OK","result":{"name":"hello", "name1":""}}
```

2.2Get

示例如下：

POST /interfaceName HTTP/1.1\r\n
Content-Type: application/xml\r\n
Content-Length: xxx\r\n
\r\n
{"param1": 1,"param2":2}

成功响应：

HTTP/1.1 **200** OK\r\n
…
\r\n

失败响应：

HTTP/1.1 **400** Method Not Allowed\r\n
…
\r\n

3 接口详细描述

3.1握手消息

接口名称 √	握手	
调用名称	/hello	
接口提供	BUBICHAIN	
接口调用		
功能描述	调用此消息获取 BUBICHAIN 的服务器信息	
请求参数		
名称	**类型（单位）**	**说明**
返回结果		
名称	**类型（单位）**	**说明**
bubi_version	string	布比版本
ledger_version	uint32	账簿版本
overlay_version	uint32	通讯版本
current_time	string	当前时间

调用示例

HTTP 请求内容:

```
POST / hello HTTP/1.1[\r\n]
Content-Type: application/json[\r\n]
...[\r\n]
[\r\n]
```

响应内容:

```
{
    "bubi_version":"2.0.0.0",//布比版本号
    "current_time":"2016-11-21  04:52:04.564353",//当前时间（linux 时间戳）
    "ledger_version":"2000",//区块版本
    "overlay_version":"1000"//通信版本
}
```

3.1生成账号地址以及密钥

接口名称 √	生成帐号	
调用名称	/createAccount	
接口提供	BUBICHAIN	
接口调用		
功能描述	调用此消息请求服务器生成账号地址以及密钥，也可以按照 **3.2** 的规则在本地生成。 该接口只是为用户生成了一对公私钥以及一个地址，要把该帐号真正意义上存入系统，还需要调用创建交易的接口（操作类型为创建帐号）。	
请求参数		
名称	**类型（单位）**	**说明**
返回结果		
名称	**类型（单位）**	**说明**
error_code	number	参考错误码说明
result	object	对象
address	string	Base58 编码
private_key	string	Base58 编码
public_key	string	Base58 编码

调用示例

HTTP 请求内容:

POST / createAccount HTTP/1.1*[\r\n]*

Content-Type: application/json*[\r\n]*

...*[\r\n]*

[\r\n]

响应内容:

```
{
"error_code":0,
    "result":{
        "address":"bubiV8i6mtcDN5a1X7PbRPuaZuo63QRrHxHGr98s",//账号地址
        "private_key":"privbwawzANVvZqh1auFef6NnD2sUvgVkahSN9hd2bdcY3EqGjpv6AXT",//账号密钥
         "public_key":"7oWXBqfJ8spKeSxQuBCQ5ZK4YLCiFUpNd25oV9ttGLtA"  //账号公钥
         }
}
```

3.2账号地址和公钥密钥生成规则

接口名称 √	生成帐号
调用名称	无
接口提供	无
接口调用	

用 ED25519 随机生成 32 字节私钥 P 和 32 字节公钥 Q。

- **遵循以下步骤生成布比私钥**

1. 将 3 字节前缀和 1 字节版本号 0XDA379F01 加到 P 前面，1 字节压缩标志添加到 P 后面即 M=0XDA379F + 0X01 + P+0X00
2. 将 M 用 SHA256 计算两次取前 4 字节，即 Checksum=SHA256(SHA256(M)) 的前 4 字节
3. 将 Checksum 的前四字节加到 M 后面，即 S=M+Checksum
4. 对 S 进行 Base58 编码即得到布比私钥。privxxxxxxxxxxxxxxxxxxxxxxxxx

- **遵循以下步骤生成布比公钥**

将 Q 进行 Base58 编码得到布比公钥

- **遵循以下步骤生成布比地址**

1. 在 Q 前面加 4 字节前缀和 1 字节版本号。即 M=0XE69A73FF+0X01+Q
2. 对 M 进行 RIPEMD160 算法得到 20 字节的 N，即 N = RIPEMD160（M）
3. 对 N 进行两次 SHA256 算法取前四字节，即 Checksum=SHA256(SHA256(N))的前 4 字节
4. 将 Checksum 加到 N 后面，即 S=N+Checksum
5. 对 S 进行 Base58 编码即得到布比地址 BUBixxxxxxxxxxx

3.3交易基本介绍

3.3.1 交易（tx）

一个交易的基本结构如下

```
"transaction_json":{
    "source_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//交易发起者,手续费支付者
    "fee":10000, //交易发起者为这笔交易支付的手续费,，若 n 个操作,则最低为 n*base_fee(在配置文件中可配置)
     "sequence_number":12,//可选，值必须是发起者账号最大交易序号+1,账号交易序号可通过获取账号信息接口获得
      "metadata":"",//可选,，值必须是 16 进制，长度为偶数
      "time_range":{//可选，最近一次区块关闭时间必须大于 min_time，小于 max_time
          "min_time":0, //linux 时间戳
          "max_time":0 //linux 时间戳
       },
       "operations":[
         {
```

```
            "type":0,//操作类型，根据操作类型值不同 operation 其他字段会有所变化
             "metadata":"",//可选,，值必须是 16 进制，长度为偶数
             "source_address":"" //可选，默认是交易发起者
                。。。//根据操作类型所定
          }
          ]
}
```

交易是布比系统中基本的执行单位。一个交易可以有 1 个或多个操作，但是最多不能超过 1000 个。一个交易是一个原子性质的操作，里面的 operation 要么全都执行，要么全不执行。
Operation 主要有创建帐号、发行资产、转账等。

3.3.2 操作（operation）

```
 "operations":[
        {
            "type":0,//操作类型，根据操作类型值不同 operation 其他字段会有所变化
             "metadata":"",//可选,，值必须是 16 进制，长度为偶数
             "source_address":"" //可选，默认是交易发起者
                。。。//根据操作类型所定
          }
]
```

每个 operation 都共有的基本 3 个字段：metadata、type、source_address。其中根据 type 的值不同 operation 的字段也会有所不同如 type 为 0(创建账号)时会多出两个必填字段 dest_address 和 init_balance

3.4交易请求方式

3.4.1 本地签名再提交服务器验证

推荐生产应用调用该接口

3.4.1.1获取未签名 blob

接口名称 √	获取交易 BLOB
调用名称	/getTransactionBlob
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息获取交易 BLOB **此接口要和[3.4.2]配合使用。**
调用示例	

HTTP 请求内容:

POST / getTransaction_blob HTTP/1.1*[\r\n]*

Content-Type: application/json*[\r\n]*

...*[\r\n]*

[\r\n]

```
{
    "source_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//交易发起者,手续费支付者
    "fee":10000, //交易发起者为这笔交易支付的手续费,，若 n 个操作,则最低为 n*base_fee(配置文件中可配)
     "sequence_number":12,//值必须是发起者账号最大交易序号+1,账号交易序号可通过获取账号信息接口获得
      "metadata":"",//可选,，值必须是 16 进制字符串，长度为偶数
      "time_range":{//可选，最近一次区块关闭时间必须大于 min_time，小于 max_time
          "min_time":0, // linux 时间戳
          "max_time":0 //linux 时间戳
       },
       "operations":[
         {
            "type":0,//操作类型，根据操作类型值不同 operation 其他字段会有所变化
             "metadata":"",//可选,，值必须是 16 进制字符串，长度为偶数
             "source_address":"" //可选，默认是交易发起者
               。。。//根据操作类型所定
          }
         ]
}
```

响应内容:

```
{
"error_code":0,//错误码
      "result":{
        "transaction_blob":"xxxxxxxxxxxxxxxxxxxxx",//交易请求序列化后的值
        "hash":"xxxxxxx"//交易请求的 hash 值
      }
}
```

3.4.1.2提交签名后 blob

接口名称 √	创建账户、包括发行资产、转账等
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息创建交易 transactionBlob 是通过接口 getTransactionBlob 返回的。 signature 是需要自行计算的。计算方法：

	1、对 transactionBlob 进行 16 进制解码。得到字节序列 A。 2、用交易发起者的私钥对 A 进行签名，得到 B 3、对 B 转换为 16 进制字符串，得到字符串 C。 4、signature 的值用 C 即可。 **此接口要和[3.4.1]配合使用。**

调用示例

HTTP 请求内容:

POST / submitTransaction HTTP/1.1*[\r\n]*

Content-Type: application/json*[\r\n]*

...*[\r\n]*

[\r\n]

```
{
"items":[
          {
            "signatures":[
             {
               "sign_data":"",//对序列化的交易请求进行签名后的值
               "public_key":"abc"//用于签名密钥对应的公钥
             }
             ],
            "transaction_blob":"" //交易请求序列化后的值
          }
        ]
}
```

响应内容:

```
{
  "success_count":1,//请求被接收数量
  "results":[
     {
       "error_code":0,//错误码，0 代表成功
       "hash":"xxxxxxxxxxxxxxxxx"//交易请求的 hash 值
     }
      ]
}
```

3.4.2 提交服务器签名并验证

推荐测试使用

接口名称 √	创建账户、包括发行资产、转账等
调用名称	/submitTransaction

接口提供	BUBICHAIN
接口调用	
功能描述	与发送请求获取交易 blob 本地签名再提交方法服务器验证不同的是改请求是把交易请求+密钥两者一同提交上去，由服务器进行代签并验证

调用示例

HTTP 请求内容:

POST / submitTransaction HTTP/1.1*[\r\n]*

Content-Type: application/json*[\r\n]*

...*[\r\n]*

[\r\n]

```
{
   "items":[
      {
         "private_keys":[
            "privC1CCDoFh9kfbKRHf9PZMPzH1N5vkBZWdbJxnkiZji7UfZsSXvUxU" //密钥
         ],
         "transaction_json":{
              "source_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//交易发起者,手续费支付者
              "fee":10000, //交易发起者为这笔交易支付的手续费，，若 n 个操作,则最低为 n*base_fee(配置文件中可配)
              "sequence_number":12,//值必须是发起者账号最大交易序号+1,账号交易序号可通过获取账号信息接口获得
              "metadata":"",//可选，值必须是 16 进制字符串，长度为偶数
              "time_range":{//可选，最近一次区块关闭时间必须大于 min_time，小于 max_time
                   "min_time":0, // linux 时间戳
                    "max_time":0 //linux 时间戳
                },
               "operations":[
                {
                      "type":0,//操作类型，根据操作类型值不同 operation 其他字段会有所变化
                      "metadata":"",//可选,，值必须是 16 进制字符串，长度为偶数
                      "source_address":"" //可选，操作源账号默认是交易发起者
                    。。。//根据操作类型所定                    }
            ]
         }
      }
   ]
}
```

响应内容:

```
{
  "success_count":1,//请求被接收数量
  "results":[
```

```
    {
      "error_code":0,//错误码，0 代表成功
      "hash":"xxxxxxxxxxxxxxxx"//交易请求的 hash 值
    }
    ]
}
```

3.5操作介绍(operation)

3.5.1 创建帐号

接口名称 √	创建账户
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息创建交易-创建帐号类型。 1、系统中创建任何帐号都必须由一个源帐号来创建，创建新账号的过程中消耗源帐号的布币。 2、系统初始有 1 个创世帐号，该帐号的公钥、地址、私钥是永久固定的，详见下面请求示例，其余一切新帐号的布币都来源于此。 若交易创建成功目标账号信息可通过查询账号信息查询到

参数说明

```
"operations":[
      {
        "type":0,//操作类型 0 为创建账号操作
        "dest_address":"bubiV8i6mtcDN5a1X7PbRPuaZuo63QRrHxHGr98s",//目标地址
        "init_balance":1001000000,//转移给目标账号的布币数量,必须大于 base_reserve（配置文件中可配）
  "account_metadata":"",//可选，设置目标账号的 account_metadata，默认为空，值必须是 16 进制字符串，长度为偶数

        "threshold":{//可选
          "master_weight":1,//可选，若不设置初始值为 1，值范围是 0-255
          "low_threshold":1, //可选，若不设置初始值为 1，值范围是 0-255
          "med_threshold":1, //可选，若不设置初始值为 1，值范围是 0-255
          "high_threshold":1//可选，若不设置初始值为 1，值范围是 0-255
        },
        "signers":[//可选,设置账号的权重
          {
            "address":"abcd1",//账号地址
            "weight":10 //该账号的权重，范围是 0~255
```

```
            }
        ]
    }
]
```

3.5.2 发行资产

接口名称 √	发行资产
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息创建交易-发行 发行资产一般是给商家添加资产的的过程。 交易若创建成功调用查询账号信息查询操作源账号信息可看到已发行的资产

参数说明

```
{
    "operations":[
        {
            "type":2,//操作类型 2 为发行资产操作
            "asset_type":1,//可选,资产类型，0 是布比，1 是用户发行的资产，2 是唯一资产 ,发行资产值只能是 1
            "asset_issuer":"",//发行商地址，这里必须与操作源账号一致
            "asset_code":"",//资产 id，长度范围是 1~64
            "asset_amount":10,//发行的数量,必须大于 0
            "details":[//可选，资产详细信息
                {
                    "amount":1,//资产数量,details 数组 amount 总和必须等于 asset_amount
                    "start":1451535975,//开始时间，若不填写 details，值默认为 0（linux 时间戳）
                    "length":31536000,//有效时间长度，若不填写 details，值默认为-1，-1 代表未初始化
                    "ext":"" //有效长度范围 0~64,若不填写 ext，值默认为空
                }
            ]
        }
    ]
}
```

3.5.3 转账

接口名称 √	转账
调用名称	/submitTransaction

接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息创建交易-转账 1、转账中如果未指定 details，系统按照**既定顺序**自动选择哪些资产。排序规则是先按照过期时间降序再按照扩展字段升序，即越接近有效期的资产越靠前。 优先顺序 快到期的>永久的>未初始化的 如果有效期相同，按照 ext 字段从小到大的顺序 2、已经过期的资产无法被用来转账，未初始化的资产、初始化过且未过期的资产（含永久）均可以转移。

参数说明

```
{
    "operations":[
        {
          "type":1,//操作类型 1 为转账操作
          "asset_type":1,//资产类型，0 是布比，1 是用户发行的资产
         "dest_address":"bubiV8i6mtcDN5a1X7PbRPuaZuo63QRrHxHGr98s",//目标地址
          "asset_amount":10,//发行的数量,必须大于 0
          "asset_issuer":"",//发行商账号地址，若 asset_type 为 0，删除该字段，若 asset_type 为 1，该字段必选
          "assset_code":"",//资产 id，若 asset_type 为 0，，删除该字段，若 asset_type 为 1，该字段必选
           "details":[//可选，资产详细信息
             {
               "amount":1,//资产数量,details 数组 amount 总和必须等于 asset_amount
               "start":1451535975,//开始时间，若不填写 details 默认为 0（linux 时间戳）
               "length":31536000,//有效时间秒数，若不填写 details 默认为-1，-1 代表未初始化
               "ext":"" //有效长度范围 0~64
             }
          ]
        }
    ]
}
```

3.5.4 初始化转账

接口名称 √	初始化转账
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息创建交易-初始化转账 所谓初始化转账是指，将**自己拥有的未初始化的**资产初始化，且转移给某人。

	1：用户只能初始化自己拥有的且**尚未初始化的**资产 2：资产一旦初始化，无法再次初始化。detail 的 length 字段指明了资产是否初始化。-1 是未初始化，0 是永久，大于 0 是有效期秒数。 3：初始化转账操作包含初始化(发行资产)和转移两个动作 4: 无法初始化转帐给自己 5:details 可选，如果未填写，默认将资产初始化为【有效期为永久，扩展字段为空】。如果填写了 details，要求其列表 amount 之和等于 asset_amount。

参数说明

```
{
    "operations":[
        {
            "type":5,//操作类型 5 为初始化转账操作
            "asset_type":1,//可选，资产类型，初始化资产只能初始化用户发行的资产，所以值只能为 1
            "dest_address":"bubiV8i6mtcDN5a1X7PbRPuaZuo63QRrHxHGr98s",//目标地址
            "asset_amount":10,//发行的数量,必须大于 0
            "asset_issuer":"",//资产发行商，必须是账号地址
            "assset_code":"", //资产 id，范围在 1~64
             "details":[//可选，资产详细信息
                {
                    "amount":1,//资产数量,details 数组 amount 总和必须等于 asset_amount
                    "start":1451535975,//开始时间，若不填写 details，值默认为 0，值必须大于等于 0（linux 时间戳）
                    "length":31536000,//有效时间秒数 ,若不填写 details ,值默认为 0， 0 代表永久 ,如果设置成 0,start
强强制设成 0，值范围大于等于-1
                    "ext":"" //有效长度范围 0~64，若不填写 details，值默认为空
                }
            ]
        }
    ]
}
```

3.5.5 发行唯一资产

接口名称 √	发行唯一资产
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	1、 发行方+唯一资产代码在布比系统网络中唯一 2、 唯一资产代码长度限制最长 64 字节 3、 发行方、代码、描述都不能为空 交易若创建成功调用查询账号信息查询操作源账号信息可看到已发行的唯一资产

参数说明

```
{
    "operations":[
        {
            "type":7,//操作类型 7 为发行唯一资产
            "asset_issuer":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy", //发行商账号地址，必须跟操作源账号地址一致
            "asset_code":"3",//资产 id，字符串长度范围是 1~64
            "asset_detailed":"0531"//值必须是 16 进制字符串，长度为偶数
        }
    ]
}
```

3.5.6 转移唯一资产

接口名称 √	转账-唯一资产
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	1、 转账不用填唯一资产描述，唯一资产只需要资产代码+发行商 2、 转账需目标地址

参数说明

```
{
    "operations":[
        {
            "type":8,//操作类型 8 为转移唯一资产
             "dest_address":"bubiV8i6mtcDN5a1X7PbRPuaZuo63QRrHxHGr98s",//目标地址
            "asset_issuer":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy", //发行商账号地址，必须跟操作源账号地址一致
            "asset_code":"3"//资产 id，字符串长度范围是 1~64
        }
    ]
}
```

3.5.7 存证

接口名称 √	存证
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	1、 每一份原声明的 id 在账户中都是唯一的 2、 追加声明可以重复 3、 要对特定事件追加声明必须要有这特定事件的原声明存在 4、 声明里属性声明地址是可选项，填了就是追加声明，对某地址的某个 id 号进行追加声明，不填就是交易发起者的原声明

参数说明

```
"operations":[
    {
      "type":9, //操作类型 9 为存证
      "record_id":"3",//存证 id，字符串长度范围在 1~64
      "record_ext":"0105",//值必须是 16 进制字符串，长度为偶数
       "record_address":"" //可选，原声明的账号地址,若填写该字段，就是追加声明，若不填写，就是原声明，追加声明必须要有原声明存在
    }
  ]
```

3.5.8 设置账户属性

接口名称 √	设置账户属性
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	设置账户属性

参数说明

```
"operations":[
  {
      "type":4, //操作类型 4 为设置账户属性
    "threshold":{//可选
      "master_weight":1, //可选，设置账户密钥的权重值，值范围是 0-255
      "low_threshold":0,//可选，设置低权限操作的权重门限值，值范围是 0-255
      "med_threshold":0,//可选，设置中权限操作的权重门限值，值范围是 0-255
      "high_threshold":0,//可选,设置高权限操作的权重门限值，值范围是 0-255
      " metadata_version":1,// 可选，如果设置了，下面的 metadata 参数必须设置，而且 metadata_version 和
```

帐号现有的 metadata_version 必须相同。帐号现有的 metadata_version 从获取账户信息接口可以查到。

```
            " metadata":"0123456789abcdef"//可选，值必须是 16 进制字符串，长度为偶数
        },
        "signers":[
            {
                "address":"abcd1",//账号地址
                "weight":"0"//账号权重，范围是 0~255，0 代表删除
            }
        ]
    }
]
```

3.5.9 供应链

接口名称 √	供应链操作
调用名称	/submitTransaction
接口提供	BUBICHAIN
接口调用	
功能描述	1、 在交易中如果有供应链的操作，那交易的操作数量只能 1 个 2、 可以没有 inputs，但不能没 outputs 调用此消息创建一个供应链操作

参数说明

```
"operations":[
    {
        "type":6, //操作类型 6 为设置供应操作
        "inputs":[//可选
            {
                "hash":"55a5955dcaeafea474c416dfe732d3411b11c3686367987eaa9f58290628fe6c",//交易中
outputs 的 address 是这笔交易操作源账号地址的交易的 hash
                "index":2,// 操作账号地址在 outputs 中的序号(从 0 开始算)
                "metadata":""//可选，值必须是 16 进制字符串，长度为偶数
            }
        ],
        "outputs":[//输出的地址,为输出地址的账号创建供应操作填写 inputs 所用
            {
                "address":"bubiV8hUYiiuBW47n56MW6LWczMn2T23U6nGkDSV",//账号地址
                "metadata":"12abef"//可选，值必须是 16 进制字符串，长度为偶数
            }
        ]
```

```
    }
  ]
```

3.6获取账户信息

接口名称 √	获取账户信息	
调用名称	/getAccount	
接口提供	BUBICHAIN	
接口调用		
功能描述	调用此消息查看账户详细信息	
请求参数		
名称	**类型（单位）**	**说明**
address	string	账户地址
返回结果		
名称	**类型（单位）**	**说明**
调用示例		

HTTP 请求内容:

GET / getAccount?address=bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy HTTP/1.1[\r\n]
Content-Type: application/json[\r\n]
...[\r\n]
[\r\n]

响应内容:


```
{
    "error_code":0,//错误码
      "result":{
        "address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//账户地址
        "assets":[ ],//账户资产列表
         "assets_unique":[ ],//账户唯一资产列表
        "balance":99999787557358000,//账户布比余额
        "hash":"ccf9e1366efca5d9c1e2759948b062da74c2ceafa4f7ccfb049c5b5396ec8baa",//账户 hash
        "last_close_time":1479728015,//最新一次区块生成时间（linux 时间戳）
        "metadata":"",//16 进制字符串
        "metadata_version":1,//metadata 版本号
        "previous_ledger_seq":1504,//该账号最新一次交易所属区块高度
        "previous_tx_hash":"9565f901bd0ea54a3bd19e38ec784ea84e70571c3891c2a918e0fe20562b4154",//该账号最新一次交易的 hash
```

```
            "signers":[],//设置的联合签名列表
            "threshold":{
                "high_threshold":1,// 账户密钥的权重值
                "low_threshold":1, //   低权限操作的权重门限值
                "master_weight":1, //   中权限操作的权重门限值
                "med_threshold":1 //   高权限操作的权重门限值
            },
            "tx_seq":4294967569  // 该账号的交易序号，初值是生成该账号的交易所在区块的区块号乘 2 的 32 次方
        }
}
```

3.7获取交易信息

接口名称 √	获取交易信息
调用名称	/getTransactionHistory
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息查询交易结果 如果返回 error_code 为 0，表示查询成功，否则表示该交易不存在。 输入参数用 hash 或 address。 如果用 hash 表示根据交易的 hash 查询该交易。 如果用 address 表示根据帐号查询与之相关的交易。 beginTime 和 endTime 是可选的过滤条件，单位是 linux 时间戳*1000000. 例如 1446015556000000 如果同时用了 hash 和 address 两个参数，则根据 hash 查询该交易,此时 address 和 begintTime 和 endTime 均不生效 返回的 transactions 数组中,每一个对象的 error_code 代表该交易成功与否.0 为成功，其他值对应系统统一错误代码.

请求参数

名称	**类型（单位）**	**说明**
hash	String	可选，根据该交易的 hash 查询交易。
address	String	可选，根据帐号查询交易。
begin_time	Uint64	可选，开始时间
end_time	Uint64	可选，结束时间
start	Uint32	可选，查询开始位置(默认从 0 开始)
limit	Uint32	可选，查询条数(默认 20 条)
Order	string	值是 desc 或 asc

返回结果		
名称	**类型（单位）**	**说明**
error_code	number	参考错误码说明
result	object	对象
transactions	string	交易列表

调用示例

HTTP 请求内容:

GET / getTransactionHistory?hash=xxxxxxxxxxxxxxxxxxxxx

GET / getTransactionHistory?address=xxxxxxxxxxxxxxxxxxxxx

HTTP/1.1[\r\n]

Content-Type: application/json[\r\n]

...[\r\n]

[\r\n]

响应内容:


```
{
   "error_code":0,
   "result":{
         "total_count":1,
         "transactions":[
            {
               "close_time":1479692280090402,//交易完成时间，与所属区块关闭时间一致（linux 时间戳）
               "error_code":0,//交易执行结果
               "fee":2000,//这笔交易设置的手续费
               "hash":"9565f901bd0ea54a3bd19e38ec784ea84e70571c3891c2a918e0fe20562b4154",//交易的 hash 值
               "ledger_seq":1054,//这笔交易所属的区块的区块高度
               "metadata":"",//16 字节字符串
               "operations":[
                  {
                     "type":0   //operation 根据交易的操作类型不同返回的字段会有所不同
                     。。。。。
                  }
               ],
               "sequence_number":4294967569,  //交易序号，在交易发起者中该序号是唯一
               "signatures":[
                  {
                     "address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//对这笔交易签名的账号地址
                     "public_key":"FYxXYJCdTvGiU2LMWkqmLVkGWLDf9RCguh1fdJ9Kyp3j",//对这笔交易签名的账号公钥
```

```
                "sign_data":"5fb0ff91f6b5fd73baace53ef3c97e612c78976231889daf35b83b29d053fa9f3
b833ad10446f70fd201e735fb7f5c60742a6a94a5ec17839f9ae65e87a3800f" //对这笔交易签名后的消息
                }
            ],
            "source_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy" //交易发起者地址
        }
    ]
  }
}
```

3.8获取区块信息

接口名称 √	获取区块信息	
调用名称	/getLedger	
接口提供	BUBICHAIN	
接口调用	钱包 APP	
功能描述	调用此消息创建账户	
请求参数		
名称	**类型（单位）**	**说明**
seq	number	可选，区块高度，不填写默认返回最高区块高度信息
返回结果		
名称	**类型（单位）**	**说明**
error_code	number	参考错误码说明
result	object	对象
调用示例		

HTTP 请求内容:

GET / getLedger?seq=2 HTTP/1.1[\r\n]
Content-Type: application/json[\r\n]
...[\r\n]
[\r\n]

响应内容:


```
{
   "error_code":0,//错误码
      "result":{
         "account_hash":"a8de93022a3b3055e6bb92fd28c15c0c0b55a7c8d907ac9dd26914aa30bd7c35",//
账户树的 hash
         "base_fee":1000,//该区块每个操作需要的手续费
         "base_reserve":10000000,//每个账号最低基本余额
         "close_time":1479785737116135,//区块关闭时间（linux 时间戳）
         "consensus_value":{//共识消息
            "hash_set":"d123b66a431f7cf3c4a857b27ddb99a9393def657e917548febe5bcab8c06afd"
         },
         "hash":"cc00e312ee59d7d5165ef6bb137137f4adbb70b68a1ddfa918270dab687635b8",//区块 hash
         "ledger_seq":1568,//区块高度
         "ledger_version":2000,//区块版本
         "phash":"5d43d203e27acb8d28a4f3177ce66b4dd7e8808cc8aea999dd3c799282b901dc",//上一个区
块 hash
         "tx_count":403,//交易总数,这里交易总数指从第一个区块开始到现在这个区块所有交易的总和
         "txhash":"e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"//当前区块
交易树的 hash
      }
   }
```


3.9查询唯一资产

接口名称 √	获取唯一资产流转信息	
调用名称	/ getUniqueAsset	
接口提供	BUBICHAIN	
接口调用		
功能描述	调用此消息查询唯一资产流转记录	
请求参数		
名称	**类型（单位）**	**说明**
asset_code	string	唯一资产代码
asset_issuer	string	发行商地址
start	string	开始位置，默认从 0 开始

limit	String	显示条数，默认从 0 开始
order	String	排序方式，根据区块高度进行比较（desc，asc）

返回结果

名称	类型（单位）	说明
error_code	number	参考错误码说明
result	object	对象

调用示例

HTTP 请求内容:

GET / getUniqueAsset?asset_code=xx&asset_issuer=xxx

HTTP/1.1[\r\n]
Content-Type: application/json[\r\n]
...[\r\n]
[\r\n]

响应内容:


```
{
    "error_code":0,
        "result":[
            {
                "from_address":"bubiV8iEL9xDrdAKk8ZstHfZoAfPMFTL8n4rgs4B",
                "to_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
                "tx_hash":"5c5fcee930f85c5c1d8a2b539094f4183a13c31e972c6236f75f165
725e77466"
            },
            {
                "from_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
                "to_address":"bubiV8iEL9xDrdAKk8ZstHfZoAfPMFTL8n4rgs4B",
                "tx_hash":"dd8590f5d49463b33cfed7fccc9b39a4ee313d6f316ce9975a09d37
d7be23e0b"
            },
            {
                "from_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
                "to_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
                "tx_hash":"71db3ba6b3d3505c5369da03b938c880b387c1a5bb6c8f2bd6a5bd4
d031ea6d9"
            }
        ]
    }
```


返回结果是交易的流转记录，排序默认是按照交易生成时间排序，从最新到最旧，最后一笔 from_address 等于 toaddress

的是发行唯一资产记录

3.10 查询存证

接口名称 √	获取区块信息
调用名称	/ getRecord
接口提供	BUBICHAIN
接口调用	
功能描述	调用此消息查询存证 当不填写 record_participant 时代表只查询出原声明 当 record_participant 值为 all 时查询出原声明以及所有对该原声明的追加声明 当 record_participant 值为账号地址时查询出原声明以及该账号对原声明的追加声明

请求参数

名称	**类型（单位）**	**说明**
id	string	可选，声明的 id
record_participant	string	可选，参与者账号，值(账号地址，all，空)
record_address	string	可选，原声明账号
start	Uint32	可选，查询开始位置(默认从 0 开始)
limit	Uint32	可选，查询条数(默认 20 条)
order	string	可选，排序方式，根据区块高度进行比较（desc,asc）

返回结果

名称	**类型（单位）**	**说明**
error_code	number	参考错误码说明
result	object	对象

调用示例

HTTP 请求内容:

GET /http://127.0.0.1:19334/getRecord?id=3&record_participant=xxxx&record_address=xxxx
HTTP/1.1[\r\n]
Content-Type: application/json[\r\n]
...[\r\n]
[\r\n]

响应内容:


```
{
    "error_code":0,
      "result":[
        {
```

```
            "record_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
            "record_ext":"Be 3",
            "record_id":"3",
            "record_participant":"bubiV8i5pfXg3dsDCj21mRtkHDJbtqccCouS4EBe",
            "tx_hash":"2a8b4a2952c784306ebc3a11b9696f76214190a66a5406581228f180e6f39105"
        },
        {
        "record_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
            "record_ext":"Be 2",
            "record_id":"3",
            "record_participant":"bubiV8i5pfXg3dsDCj21mRtkHDJbtqccCouS4EBe",
            "tx_hash":"dc21879812866c59fe914f7691afaaf5347cf36c2f8555964ee6f9b3aeeb0706"
        },
        {
            "record_address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
            "record_ext":"Be 1",
            "record_id":"3",
            "record_participant":"bubiV8i5pfXg3dsDCj21mRtkHDJbtqccCouS4EBe",
            "tx_hash":"bb056ffa5c69651805bb16805fbd705fd75195d6b049ae93103d9a9f7f4e0836"
        },
        {
            "record_address":"",
            "record_ext":"hello",
            "record_id":"3",
            "record_participant":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",
            "tx_hash":"54fe5a87aea096cce737fb53d437cdb6b57d70e1908f0f715047b9f005c2cbbc"
        }
    ]
}返回结果中数组默认排序是从最新到最旧。最后一个是原声明，默认显示前 20 条声明记录
```


3.11 供应链溯源

接口名称 √	获取区块信息	
调用名称	/getSources	
接口提供	BUBICHAIN	
接口调用	钱包 APP	
功能描述	调用此消息创建账户	
请求参数		
名称	**类型（单位）**	**说明**
hash	string	要追踪的 hash
depth	int	最大深度

返回结果

名称	类型（单位）	说明
error_code	number	参考错误码说明
result	object	对象

调用示例

HTTP 请求内容:

```
GET / getLedger?hash=xxxxxxxx&depth=3   HTTP/1.1[\r\n]
Content-Type: application/json[\r\n]
...[\r\n]
[\r\n]
```

响应内容:

```
{
   "error_code":0,
      "result":{
         "address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//交易 A 的交易发起者
         "error_code":0,//交易 A 的错误代码
         "from":[//交易 A 中 input 的交易
            {
               "address":"bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy",//交易 B 的交易发起者
               "error_code":0,//交易 B 的错误代码
               "from":null,//交易 B 没有 input
               "hash":"4c99da180637b33b6ebf15bccd73d1de323782b37f96a8e91b593cd311a7369d",//交易 B
的 hash 值
                "index":0,//交易 B 的第 0 个输出作为了交易 A 的输入
                "input_meta":"2222",//交易 A 在这个 input 的 metadata
                "output_meta":"1111"//交易 B 的第 0 个输出的 metadata
                 }
            ],
         "hash":"ad118f34c466f442921aee331d83e1249d2020afe0d6574fe51c75e9362d4056"
      }
}
```

说明：

上述结果包含 2 个交易，

交易 A 的 hash="ad118f34c466f442921aee331d83e1249d2020afe0d6574fe51c75e9362d4056"

交易 B 的 hash="4c99da180637b33b6ebf15bccd73d1de323782b37f96a8e91b593cd311a7369d"

交易 A 的某个个输入来自交易 B 的某个输出。

由于交易 A 是溯源开始的交易，所以没有其他交易以 A 作为输入，所以交易 A 中没有 output_meta 和 index。

而交易 A 的 input_meta 是可以有多个的，所以放在了 from 中。

4 错误代码

以下错误码在整个系统的所有接口中通用。

错误码	含义	备注
0	成功	
1	内部错误	底层问题
2	参数错误	调用接口用的参数不对。参数缺失或参数范围超过规定。
3	目标已经存在	
4	目标不存在	
90	公钥非法	
91	私钥非法	
92	资产非法	
93	验证签名失败	
94	地址非法	
95	不在时间范围内	
96	没有共识	
97	交易中缺少缺少操作	
98	交易中包含的操作数量超过限制	
99	交易的序号非法	
100	可用内置币余额不足	
101	目标地址不能等于源地址，非法	
102	目标帐号已经存在	
103	目标账户不存在	
104	可用资产余额不足	
111	提供的手续费不足	
112	交易提交过早	
113	交易提交过晚	
114	服务器收到的交易数过多,正在处理	
120	权重值无效	
130	输入不存在	仅用于供应链场景中，input 不存在
131	输入非法	仅用于供应链场景中，input 提供的 hash 和 index，不能作为 input 使用，可能的原因 1：上个 tx 的类型不是 production 2：上个 tx 不是成功状态
132	非供应链类型交易	溯源过程中,要溯源的交易并不是供应链类型交易
144	账户的 metadata 版本号错误	

5 系统常量

5.1 操作类型码

操作类型码	名称	备注
0	创建帐号	用来新建帐号
1	转账	调用此接口可以将布币或自定义资产转给另一个帐号
2	发行资产	发行资产
3	保留	保留，以后使用
4	设置帐号属性	设置帐号的签名者列表，各个类型的权重值
5	初始化转账	将尚未初始化的资产初始化并转给另一个帐号
6	供应链记录	提交一个供应链的操作
7	发行唯一资产	发现唯一资产
8	转账唯一资产	调用此接口可以将唯一资产转给另外一个账号
9	存证	调用此接口做一份存证声明

5.2 创世帐号

地址:bubiV8i2MLZd5ahDGay6oAZHiMyYNUkJfSiTAmJy
私钥:privC1CCDoFh9kfbKRHf9PZMPzH1N5vkBZWdbJxnkiZji7UfZsSXvUxU
公钥:FYxXYJCdTvGiU2LMWkqmLVkGWLDf9RCguh1fdJ9Kyp3j

6 参考资料

列举出相关引用和参考资料，格式如下：

1. RFC793 - Transmission Control Protocol
2. RFC2616 - Hypertext Transfer Protocol -- HTTP/1.1
3. RFC2246 - The Transport Layer Security (TLS) Protocol 1.0